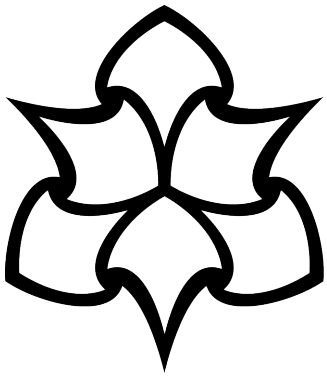


Zhao, W, Beach, TH and Rezgui, Y (2017) Efficient least angle regression for identification of linear-in-the-parameters models. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 473 (2198). ISSN 1364-5021

Efficient least angle regression for identification of linear-in-the-parameters models

Wanqing Zhao, Thomas H. Beach and Yacine Rezgui

Least angle regression (LAR), as a promising model selection method, differentiates itself from conventional stepwise and stagewise methods, in that it is neither too greedy nor too slow. It is closely related to L_1 norm optimisation, which has the advantage of low prediction variance through sacrificing part of model bias property in order to enhance model generalisation capability. In this paper, we propose an efficient least angle regression algorithm (ELAR) for model selection for a large class of linear-in-the-parameters (LIP) models with the purpose of accelerating the model selection process. The entire algorithm works completely in a recursive manner, where the correlations between model terms and residuals, the evolving directions and other pertinent variables are derived explicitly and updated successively at every subset selection step. The model coefficients are only computed when the algorithm finishes. The direct involvement of matrix inversions is thereby relieved. A detailed computational complexity analysis indicates that the proposed algorithm possesses significant computational efficiency, compared to the original approach where the well-known efficient Cholesky decomposition is involved in solving least angle regression. Three artificial and real-world examples are employed to demonstrate the effectiveness, efficiency and numerical stability of the proposed algorithm.

1. Introduction

Research into a large class of linear-in-the-parameters (LIP) models for nonlinear system identification has consistently drawn substantial interest from both academic and industrial communities [1–6], making it an important

research topic in the field of computational intelligence and machine learning. The main reason can be attributed to the fact that a variety of conventional and advanced models, networks or systems, such as polynomial models [7–9], RBF (radial basis function) neural networks [10–12], fuzzy rule-based systems [13–15] and least-squares support vector machines [16–18], can somehow be treated and trained as LIP models. The essential concept of LIP models is that they are formed by a set of linearly combinatorial model terms which are in turn nonlinearly mapped from the original model input space. For instance, a fuzzy system can be viewed as a series of linear expansion of fuzzy basis functions transformed from a model input space [13], where the claimed interpretability virtue lies in explaining the developed fuzzy system in terms of understandable IF-THEN linguistic rules, where the rule premises are employed to partition the input space into fuzzy regions in which the rule consequents are valid to describe the system's behaviour. The model terms in LIP models are also recognised as, or closely related to some well-known terminologies, e.g., monomials, basis functions, hidden nodes, fuzzy rules and support vectors, in various research contexts. Here, except for polynomial models, nonlinear parameters are often involved in the model terms for mapping the original model space into another hyperspace in order to improve model performance.

Although LIP models are widely considered to have strong approximation abilities given a complex enough model structure, challenging scientific problems arise in how to effectively and efficiently determine a proper model structure and the associated parameters, a process which is often known as model selection (aka subset selection). A simpler model structure with fewer model terms included is usually preferred from several perspectives, such as model interpretability, model sparsity and model generalisation capability. For example, a small number of fuzzy rules included in a fuzzy system would help understand the underlying system behaviours being described. Meanwhile, the associated unknown model parameters also need to be estimated. In practice, a large number of candidate model terms can usually be generated from measured data, from which the selection of a compact set of model terms that make up a LIP model with acceptable model performance is desirable. If an excessive number of model terms are adopted, an unnecessarily complex LIP model can thereby be resulted with deteriorated model interpretability and generalisation abilities.

Therefore, the model selection of LIP models aims to find a small subset of model terms for predicting system output with good accuracy. This is a very important research area and constitutes the main topic of the paper. Unsupervised learning methods are amongst earlier attempts for performing model selection, the well-known representatives being the clustering-based methods and rank-revealing or decomposition-based methods [19–24]. These methods are intrinsically fast as usually only data from input space is processed without explicitly considering the system output information. This inevitably leads to important difficulties, i.e., inaccurate models learnt from data, because feedback does not exist to assess model output using training data. It is noted though clustering methods may also employ output information in certain cases, but turning out that the corresponding computational complexities are dramatically increased and the connection of the obtained clusters to a particular LIP model is not always straightforward. Therefore, a refinement stage is required to further enhance model performance, where the so-called supervised learning is introduced. Consequently, research on accurate model selection methods is more focused on supervised learning methods.

Amongst supervised learning methods, obsolete approaches use exhaustive search to examine every size of possible combinations of model terms in the least-squares sense, which is computationally extremely expensive and even practically infeasible when a large number of terms are available [15]. Fortunately, the stepwise selection techniques [25] provide alternative approaches to significantly mitigate the computational burden at the expense of introducing locally optimal solutions. In the framework of forward stepwise selection, it performs the model selection sequentially, in that each step a new term is included into the model to maximumly improve model performance, where the model coefficients for a series of candidate models are normally computed in the least-squares sense. This means that the model residual vector obtained

after the inclusion of each new model term is always orthogonal to all the selected model terms. It still exhibits heavy computational burden behind the raw idea of the stepwise approach, as many candidate model terms and inverse operations are usually involved in the model selection process.

Nowadays, there are a few efficient model selection algorithms working on the forward stepwise principle though distinct model searching criteria are adopted. For example, despite different schemes used to deal with the involved inverse operations, the orthogonal least-squares (OLS) [26–30] and fast recursive algorithm (FRA) [3,31] were proposed to pursue the largest reduction of L_2 norm of model residuals at every model selection step. Alternatively, the correlation between the model residual resulting from each step and the remaining unselected model terms can also be used as the searching criterion instead of the residual's L_2 norm, where the candidate term giving the largest correlation is selected at the following step. In addition to the forward stepwise approaches, it may be worth mentioning that backward stepwise approaches [25] are similar techniques but working in a reverse manner (starting with the inclusion of all candidate model terms and then eliminating the most insignificant one at a time). However, the whole process can be inefficient, especially when a large number of candidate model terms are available.

It has been found that the forward stepwise approach performs greedy optimisation in the sense that each time the best model term is greedily (where the least-squares estimation is performed, resulting in the selected terms completely orthogonal to the model residual, i.e., zero correlation between them) added into the selected pool based on a number of previously chosen terms. Given this greedy optimisation methodology, it can normally find locally optimal solutions according to some searching criterion. Instead, the forward stagewise selection as a less greedy approach opposed to the forward stepwise [32,33], iteratively identifies the most correlated term with the current model residual and thereby only updates its own coefficient with an amount proportional to the corresponding correlation (but not equal to it, meaning a smaller optimisation step compared to stepwise) at each iteration step. As a result, the resultant correlation for the selected terms is non-zero and is decreased during the model selection process, until reaching zero and the least-squares solution is thus approached. The downside therefore lies in that the iterative working manner of this approach requires a large number of iteration steps (usually being considerably larger than the available number of model terms), incurring huge computational burden. It is worth noting that, to improve optimality and model generalisation performance, distinct enhancements have also been made to the forward stepwise methods based on either OLS or FRA, including the involvement of additional refinement phases (where the selected model terms are reviewed and replaced by unselected terms with larger contribution to the underlying model) and/or the regularisation techniques (where penalised cost functions are imposed on either the original or converted orthogonal space of the system) [15,34–37]. The model performance has since been greatly improved while a global optimality cannot be guaranteed, the computational burden being thereby increased due to the extra computing phases and/or (determination of) the introduced regularisation parameters.

As a trade-off between the aforementioned stepwise and stagewise approaches, the newly joined model selection approach, least angle regression (LAR) [25,32], provides another efficient model selection framework motivated from the geometrical perspective [38]. It has also been found that a direct modification of LAR can lead to the full-path of Lasso (Least absolute shrinkage and selection operator) solutions [25,32,39], where L_1 regularised cost is adopted to minimise the sum of squared residuals (SSR) and the L_1 norm of model coefficients. LAR uses a similar working manner as in stepwise while adopts the concept behind stagewise, but the key idea instead is to decrease the correlation for the selected terms at each step at a reasonably large step size that just makes the correlation for the selected terms equal to the correlation for an unselected term. The correspondingly unselected term is then added into the model and the correlations are decreased equally amongst all the selected terms during the model selection process until reaching the least-squares estimation given all available terms selected. Differing

from the greedy selection employed in the stepwise approach and the iterative optimisation (indicating high computational demand) exhibited in the stagewise approach, LAR owns both less greedy merits and the efficient stepwise selection manner (indicating less computational demand). In this respect, it might be worth mentioning that the L_2 norm regularised optimisation (aka ridge regression) can also be used for shrinking model coefficients towards zeros. However, it is unable to force coefficients to exact zeros, in order to remove the corresponding model terms from the final model. Apart from working on linear regression models, it is interesting to note that, LAR (specifically its underpinning equiangularity condition) has also been generalised in a number of different settings. For instance, it was extended to generalised linear models [40] through considering the model's differential geometrical structure. In addition, relating the correlation for various model terms to the corresponding gradient of the cost function under investigation, LAR has also recently been extended for applying on generalised linear/quasi-likelihood models [41], Cox's proportional hazard models [42] and more generally any convex cost functions [38], where ordinary differential equations (ODEs) are successively solved to infer the corresponding solution path.

In particular, regarding the greediness of stepwise algorithms, such as OLS, theoretically, they lead to the resultant model residuals being always orthogonal to the selected model terms, which means that the terms are thoroughly added into the model at each step of model selection (i.e., giving the best modification at each single step irrespective of the future effect [43]). In principle, this mechanism is aggressive and can be overly greedy as useful terms which are correlated with selected ones are unlikely to be included into the model subsequently due to the low explanatory power given on the underlying model. In this regard, stagewise methods which partially add terms into the model at some small/tiny step size at each step can also be employed to understand the greedy behaviour exhibited by the forward stepwise method (being corresponded to that of a large enough step size used in stagewise). Since the stagewise method itself is computationally very slow due to the many tiny iterative optimisation steps, thus LAR was proposed to dramatically mitigate the computational demand with the correlation between selected terms and model residuals decreases in a reasonably fast manner determined algebraically. Simulations have also been designed and conducted to verify the above points by Efron et al. [32]. It had been found that, as the model size increases, the predictive model performance (goodness of fit) of LAR was able to rise reasonably quickly, then decrease very slowly after reaching some maximum value. In contrast, the model performance of forward stepwise method can rise and decline more sharply than that of LAR. In this sense, the generalisation performance of stepwise can be better than that of LAR at the beginning of model selection just due to its greedy nature, but as more model terms were then subsequently included into the model, the situation was gradually reversed. The generalisation performance of LAR was thereafter seen better than stepwise and remained outperforming (considerably less over-fitting). These had therefore demonstrated the dangerously greedy nature of forward stepwise approach. For details, they have been well elaborated in the original LAR paper [32] and further claimed in a number of related studies thereafter [43,44].

Given the stepwise selection manner of least angle regression, there still can be substantial candidate model terms and matrix inversions involved in the model selection process, resulting in high computational demand. In this paper, we derive a new efficient recursive algorithm for solving the least angle regression, as opposed to the well-known OLS and FRA used for solving the forward stepwise regression. The correspondingly selected model terms are thereby used in the context of LIP model construction. The proposed algorithm deals with the least angle regression recursively without the direct involvement of matrix inversions. The correlations between model terms and residuals, the evolving directions and other pertinent variables are explicitly expressed and successively updated in a recursive fashion. The model coefficients are only computed at the end of the algorithm. The computational complexity of the proposed algorithm is then accurately analysed. Examples from Chaotic time-series prediction, number of sunspots forecasting, to Australian credit approval, are finally employed to demonstrate the

effectiveness, efficiency and numerical stability of the proposed approach in comparison to the original approach (where the well-known efficient Cholesky decompositions are involved).

The rest of the paper is organised as follows. Section 2 gives the mathematical formulation of LIP models and the least angle regression. The proposed algorithm and its adoption for LIP model identification are then presented in Section 3. The working principle of the algorithm and the corresponding computational complexity are also described in detail. Results from three artificial and real-world examples are given in Section 4. Finally, Section 5 concludes the paper.

2. Linear-in-the-parameters models and least angle regression

(a) Linear-in-the-parameters models

The general representation of a linear-in-the-parameters (LIP) model [1–3] can be expressed as

$$y(t) = \sum_{i=1}^{M} \varphi_i \left(\mathbf{x}(t), \mathbf{W}\right) \theta_i + e(t), \tag{2.1}$$

where $\mathbf{x}(t) = [x_1(t), \ldots, x_n(t)] \in \Re^n$ constitutes the model input vector at time instant t, $y(t)$ is the system output, $e(t)$ is the model residual, φ_i is the ith model term used to map model input $\mathbf{x}(t)$ into the corresponding hyperspace using some nonlinear parameter $\mathbf{W}$, θ_i is the associated coefficient for the i model term, M is the total number of model terms included in the LIP model. Different basis functions can be used to form model terms, here, assuming that a radial basis function (RBF) is adopted (thus to construct a RBF neural network):

$$\varphi_i(\mathbf{x}(t), \mathbf{W}) = \exp\left\{ -\frac{1}{2\sigma^2} \sum_{j=1}^{n} \left[x_j(t) - c_{i,j}\right]^2 \right\}, \tag{2.2}$$

where $c_{i,j}$ and σ represent the centre and width of the corresponding basis function. Therefore, the nonlinear parameter vector involved in all the model terms is given by $\mathbf{W} = [\mathbf{c}_1^{\mathrm{T}}, \ldots, \mathbf{c}_M^{\mathrm{T}}, \sigma]^{\mathrm{T}}$, where $\mathbf{c}_i = [c_{i,1}, \ldots, c_{i,n}]^{\mathrm{T}} \in \Re^n$ for $i = 1, \ldots, M$. The associated model coefficient vector is given by $\boldsymbol{\Theta} = [\theta_1, \ldots, \theta_M]^{\mathrm{T}}$. Initially, a large number of candidate model terms φ_i can be generated from measured data. The task of model selection is to find a subset of model terms (e.g., hidden nodes determination in RBF neural networks), say p_i, from the initial candidate model terms φ_j, to construct a parsimonious model.

It can be seen that the mathematical formulation of LIP models can be viewed as the linear combination of a series of model terms (laying the name of linear-in-the-parameters). It is capable of approximating any continuous nonlinear function $f(\cdot)$ arbitrarily well on a compact set, given enough number of model terms. For example, consider the following nonlinear autoregressive model with eXogenous inputs (NARX) [45,46], commonly seen in time-series prediction:

$$y(t) = f(\mathbf{u}(t-1), \ldots, \mathbf{u}(t-l_u), y(t-1), \ldots, y(t-l_y)) + e(t), \tag{2.3}$$

where $\mathbf{u}(t) = [u_1(t), \ldots, u_s(t)]$ (s-dimensional) and $y(t)$ are respectively the original system input and output variables, l_u and l_y are the corresponding maximal time lags for the inputs and output, $e(t)$ is the model residual, and $f(\cdot)$ is some unknown nonlinear function. The LIP model presented in (2.1) can then be utilised to approximate the unknown function $f(\cdot)$, while using $\mathbf{x}(t) = [\mathbf{u}(t-1), \ldots, \mathbf{u}(t-l_u), y(t-1), \ldots, y(t-l_y)] \in \Re^n$ ($n = l_u s + l_y$) as the model input vector. Given a total of N training patterns, substituting (2.1) into (2.3) and expressing the results in matrix form, yields

$$\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\Theta} + \mathbf{e}, \tag{2.4}$$

where $\mathbf{y} = [y(1), \ldots, y(N)]^{\mathrm{T}}$ represents the system output vector, $\boldsymbol{\Phi} = [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_M]$ ($\boldsymbol{\varphi}_i = [\varphi_i(\mathbf{x}(1), \mathbf{W}), \ldots, \varphi_i(\mathbf{x}(N), \mathbf{W})]^{\mathrm{T}}$, $i = 1, \ldots, M$) is the regression matrix, $\boldsymbol{\Theta} = [\theta_1, \ldots, \theta_M]^{\mathrm{T}}$ is the coefficient vector, and $\mathbf{e} = [e(1), \ldots, e(N)]^{\mathrm{T}}$ is the model residual vector. The objective turns out to select a number of regressors $\mathbf{p}_i$ (say $i = 1, \ldots, m$), from the initial candidate regressors $\boldsymbol{\varphi}_j$ ($j = 1, \ldots, M$), and identify the corresponding coefficient vector $\hat{\boldsymbol{\Theta}}_m \in \Re^m$.

(b) Least angle regression

Least angle regression (LAR) [25,32], as a relatively new model selection method, aims to find a parsimonious set of model terms for the best of prediction. LAR stands itself between the well-known forward stepwise regression and forward stagewise regression, while avoiding the downsides from both approaches (i.e., neither too greedy nor too tardy during the model selection process as discussed in Section 1). All the model terms are assumed to be standardised with zero mean and unit variance, while the system output has the mean subtracted.

LAR starts by finding the largest absolute correlation between the candidate model terms and the system output, resulting in the corresponding term being first selected for regression. Then, at every step, the step size (in turn determining the next model term to be added into the model) is chosen as small as possible, in such a way that the absolute correlation based on the resultant model residual, for some remaining term is just as large as the one for the selected terms (i.e., along with the so-called "least angle direction"). The corresponding model term obtained at every step is thus successively entered into the model and the selection therefore always proceeds equiangularly between all the selected terms. The inclusion of model terms can be viewed in a piecewise fashion, each resulting in a submodel contributing to the whole model. In detail, we can thus assume that the overall model output $\hat{\mathbf{y}}_k$ resulting from introducing the submodel at the kth step is given by

$$\hat{\mathbf{y}}_k = \hat{\mathbf{y}}_{k-1} + \boldsymbol{\Phi}_k \hat{\boldsymbol{\theta}}_k = \hat{\mathbf{y}}_{k-1} + \gamma_k \boldsymbol{\Phi}_k (\boldsymbol{\Phi}_k^{\mathrm{T}} \boldsymbol{\Phi}_k)^{-1} \boldsymbol{\Phi}_k^{\mathrm{T}} \mathbf{e}_{k-1}, \tag{2.5}$$

where $\hat{\mathbf{y}}_{k-1}$ and $\mathbf{e}_{k-1}$ are respectively the total model output and the overall model residual at the $(k-1)$th step, and $\boldsymbol{\Phi}_k$ and γ_k $(0 \leq \gamma_k \leq 1)$ are respectively the regression matrix and the step size at the kth step. The kth submodel is thus configured based on the previously resultant model residual $\mathbf{e}_{k-1}$. It should be noted that $\boldsymbol{\Phi}_k \in \Re^{N \times k}$ is constructed by expanding $\boldsymbol{\Phi}_{k-1} \in \Re^{N \times (k-1)}$ (involving all previously selected model terms) with the new model term $\mathbf{p}_k$ just selected at the kth step, i.e., $\boldsymbol{\Phi}_k = [\boldsymbol{\Phi}_{k-1},\ \mathbf{p}_k]$. The resultant overall model residual at the kth step can thus be obtained by

$$\mathbf{e}_k = \mathbf{y} - \hat{\mathbf{y}}_k = \mathbf{e}_{k-1} - \gamma_k \boldsymbol{\Phi}_k (\boldsymbol{\Phi}_k^{\mathrm{T}} \boldsymbol{\Phi}_k)^{-1} \boldsymbol{\Phi}_k^{\mathrm{T}} \mathbf{e}_{k-1}, \tag{2.6}$$

The overall model output at the kth step can be written as

$$\begin{aligned} \hat{\mathbf{y}}_k &= \sum_{i=1}^{k} \boldsymbol{\Phi}_i \hat{\boldsymbol{\theta}}_i \\ &= \boldsymbol{\Phi}_k \left[\begin{pmatrix} \hat{\boldsymbol{\theta}}_1 \\ \mathbf{0}_{(k-1)\times 1} \end{pmatrix} + \begin{pmatrix} \hat{\boldsymbol{\theta}}_2 \\ \mathbf{0}_{(k-2)\times 1} \end{pmatrix} + \cdots + \hat{\boldsymbol{\theta}}_k \right] \\ &= \boldsymbol{\Phi}_k \hat{\boldsymbol{\Theta}}_k, \end{aligned} \tag{2.7}$$

where $\mathbf{0}_{(k-i)\times 1}$ denotes a zero column vector with $k-i$ entries $(i = 1, \ldots, k-1)$ and $\hat{\boldsymbol{\theta}}_i$ is given as

$$\hat{\boldsymbol{\theta}}_i = \gamma_i (\boldsymbol{\Phi}_i^{\mathrm{T}} \boldsymbol{\Phi}_i)^{-1} \boldsymbol{\Phi}_i^{\mathrm{T}} \mathbf{e}_{i-1}, \quad i = 1, \ldots, k. \tag{2.8}$$

The key idea behind LAR is to find the step size γ_k at the kth step $(k = 1, \ldots, M-1)$ in order to include the $(k+1)$th model term into the piecewise model, such that the resultant absolute correlation between the selected terms $(\mathbf{p}_1, \ldots, \mathbf{p}_k)$ and the resultant error $(\mathbf{e}_k)$ is just equal to the largest absolute correlation between the remaining terms $(\boldsymbol{\varphi}_{k+1}, \ldots, \boldsymbol{\varphi}_M)$ and the resultant error $(\mathbf{e}_k)$. Using (2.6), the correlation for the existing terms in the regression matrix and for the remaining terms in the candidate pool can be respectively computed as

$$\begin{cases} \boldsymbol{\Phi}_k^{\mathrm{T}} \mathbf{e}_k = (1-\gamma_k) \boldsymbol{\Phi}_k^{\mathrm{T}} \mathbf{e}_{k-1} \Rightarrow \mathbf{p}_i^{\mathrm{T}} \mathbf{e}_k = (1-\gamma_k) \mathbf{p}_i^{\mathrm{T}} \mathbf{e}_{k-1}, & i = 1, \ldots, k; \quad (2.9) \\ \boldsymbol{\varphi}_i^{\mathrm{T}} \mathbf{e}_k = \boldsymbol{\varphi}_i^{\mathrm{T}} \mathbf{e}_{k-1} - \gamma_k \boldsymbol{\varphi}_i^{\mathrm{T}} \boldsymbol{\Phi}_k (\boldsymbol{\Phi}_k^{\mathrm{T}} \boldsymbol{\Phi}_k)^{-1} \boldsymbol{\Phi}_k^{\mathrm{T}} \mathbf{e}_{k-1}, & i = k+1, \ldots, M. \quad (2.10) \end{cases}$$

Let $\left|\mathbf{p}_i^{\mathrm{T}}\mathbf{e}_{k-1}\right| = \rho_k$, (for every $i = 1, \ldots, k$), the step size γ_k ($k = 1, \ldots, M-1$) can therefore be determined by

$$\gamma_k = \min_{i=k+1}^{M} \left[\frac{\pm\boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{e}_{k-1} - \rho_k}{\pm\boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1} - \rho_k} \right]_+, \tag{2.11}$$

where sign "+" means that only the positive values are taken into account in order to find the minimum and positive value for the assignment of γ_k, while the two signs "±" in the denominator and numerator stand for that the corresponding associated values can be taken as either the positive or the negative, simultaneously. As a result, the $(k+1)$th term $\mathbf{p}_{k+1}$ is selected as $\mathbf{p}_{k+1} = \arg\gamma_k$ and is then included to form the next selected regression pool $\boldsymbol{\Phi}_{k+1} = [\boldsymbol{\Phi}_k,\ \mathbf{p}_{k+1}]$. Noting that if all the available model terms are to be included in the regression matrix, i.e., $\boldsymbol{\Phi}_M = [\mathbf{p}_1, \ldots,\ \mathbf{p}_M]$, it takes $\gamma_M = 1$, which essentially corresponds to a full least-squares fit. To help understand the entire solution searching process, the pseudo code of LAR is summarised in Algorithm 1.

Algorithm 1 Pseudo code of least angle regression

1: Initialise $\boldsymbol{\Phi}_0 \leftarrow [\,]$, $\mathbf{L}_0 \leftarrow [\,]$ and $\hat{\boldsymbol{\Theta}}_0 \leftarrow [\,]$.
2: Assign $\hat{\mathbf{y}}_0 \leftarrow \mathbf{0}$ and $k \leftarrow 1$.
3: **while** $k \leq m$ **do**
4: Compute model error $\mathbf{e}_{k-1} \leftarrow \mathbf{y} - \hat{\mathbf{y}}_{k-1}$.
5: Compute correlation $\boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{e}_{k-1}$ for unselected model terms ($i = k, \ldots, M$).
6: Find the model term with the largest correlation $\{\mathbf{p}_k, \rho_k\} \leftarrow \arg\max_{i=k}^{M} |\boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{e}_{k-1}|$ and thus update regression matrix $\boldsymbol{\Phi}_k \leftarrow [\boldsymbol{\Phi}_{k-1},\ \mathbf{p}_k]$.
7: Perform Cholesky decomposition on $\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k = \mathbf{L}_k\mathbf{L}_k^{\mathrm{T}}$ based on previously obtained lower triangular matrix $\mathbf{L}_{k-1}$ to gain $\mathbf{L}_k$.
8: Compute $(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$, $\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$ and $\boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$, ($i = k+1, \ldots, M$), sequentially, based on $\mathbf{L}_k$ using forward and backward substitutions.
9: Compute step size γ_k using (2.11).
10: Update model coefficient $\hat{\boldsymbol{\Theta}}_k \leftarrow [\hat{\boldsymbol{\Theta}}_{k-1}^{\mathrm{T}},\ 0]^{\mathrm{T}} + \gamma_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$.
11: Update model output $\hat{\mathbf{y}}_k \leftarrow \hat{\mathbf{y}}_{k-1} + \gamma_k\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$.
12: Update $k \leftarrow k+1$.
13: **end while**
14: Output $\boldsymbol{\Phi}_m$ and $\hat{\boldsymbol{\Theta}}_m$.

Here, in the original approach, it can be found that, to avoid the inverse operation at every step of LAR, say the kth step, the computation as well as the well-known efficient Cholesky decomposition (being widely considered more efficient than orthogonal and singular value decompositions [26]) is applied on term $\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k$ (this can be performed efficiently based on the results obtained at the last selection step). Then, the corresponding model parameters $(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$ (via forward and backward substitutions), terms $\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$ and $\boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$, step size γ_k, model coefficient $\hat{\boldsymbol{\Theta}}_k$, model output $\hat{\mathbf{y}}_k$, model residual $\mathbf{e}_k$ and correlation for unselected model terms $\boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{e}_k$, are subsequently computed. The whole process is still inefficient and thus exhibits high computational demand. As opposed to the original approach, in this paper, a new efficient algorithm will be proposed in the next section without the need of matrix inversions. The whole algorithm works in a recursive fashion, the correlations and evolving directions, together with other pertinent variables being explicitly formulated and successively updated. The model coefficients only need to be computed when the model selection process finishes.

It may be worth mentioning that a modification to the LAR method can actually lead to the full-path solutions for another well-known model selection method Lasso (Least absolute shrinkage and selection operator), where a series of solutions at critical points corresponding

to the adjustion of the L_1 regularisation parameter in Lasso can be obtained. Based on the optimal Karush-Kuhn-Tucker (KKT) conditions of Lasso [47], to retrieve Lasso solutions, the essential idea thereby is to enforce that the coefficient sign of the model terms determined by LAR remains unchanged so long as they are included into the model [25,32,39]. This can be realised by removing the selected term from the model through choosing a proper step size, once its coefficient is detected to be changed if the model selection continues to proceed as in LAR. In this manner, sequential sets of Lasso solutions in different size, based on different degree of regularisation, can be achieved in an efficient way rather than solving quadratic programming (QP) problems as exhibited by the KKT optimality. Given the involvement of term removal stage, more computing steps (i.e., computational time) than that in LAR are also needed to acquire such Lasso solutions. Though LAR and Lasso were derived from distinct principles/objectives, but they do often produce similar results as claimed by Hastie et al. and Efron et al. [25,32]. Whilst the scope of this paper is centralised in the derivation of efficient LAR algorithm for identification of LIP models, the efficient realisation of Lasso is being derived and examined as part of another research studying the predictive model construction for combined sewer overflows forecasting in the urban wastewater collection environment.

3. Model selection based on efficient least angle regression

As presented in (2.4), one of the most challenging tasks involved in the LIP model construction is to determine a compact set of model terms and the associated parameters $\mathbf{W}$ and $\boldsymbol{\Theta}$. Here, assuming that RBF neural networks are adopted, a total of N candidate model terms (where $M = N$) can always be generated by using all the given training samples as potential centres (to consist $\mathbf{W}$) of basis functions, leading to a candidate selection pool $\boldsymbol{\Phi}_N = [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_N]$. The task then turns into finding the most significant terms contained in this pool to construct the final regression matrix, say $\boldsymbol{\Phi}_m$, and meanwhile determining the corresponding coefficient vector $\boldsymbol{\Theta}_m$.

(a) Efficient least angle regression

It can be shown in (2.11) that in order to compute the step size and thus to select the model terms in sequential, the correlation term $\boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{e}_{k-1}$ and the direction term $\boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}$ needs to be explicitly computed. In this paper, a new efficient least angle regression (ELAR) algorithm is now proposed to compute them by solving the least angle regression recursively.

To start, define a correlation vector $\mathbf{c}^{k)} \in \Re^{M-k}$ as for the remaining model terms in the candidate pool and the associated direction vector as $\mathbf{d}^{k)} \in \Re^{M-k}$ at the kth step, with their ith entries being respectively given as

$$\begin{cases} c_i^{k)} = \boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{e}_k, & k = 0, \ldots, M-1,\ i = k+1, \ldots, M; \quad (3.1) \\ d_i^{k)} = \boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}, & k = 1, \ldots, M-1,\ i = k+1, \ldots, M. \quad (3.2) \end{cases}$$

By using (2.9) and (2.10), the following can be recursively obtained:

$$\begin{cases} \rho_k = (1-\gamma_{k-1})\rho_{k-1}, & k = 2, \ldots, M; \quad (3.3) \\ c_i^{k-1)} = c_i^{k-2)} - \gamma_{k-1}d_i^{k-1)}, & k = 2, \ldots, M,\ i = k, \ldots, M. \quad (3.4) \end{cases}$$

where $\rho_1 = |\mathbf{p}_1^{\mathrm{T}}\mathbf{y}|$ and $c_i^{0)} = \boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{y}$, $(i = 1, \ldots, M)$. The step size in (2.11) can now be re-expressed as

$$\gamma_k = \min_{i=k+1}^{M} \left[\frac{\pm c_i^{k-1)} - \rho_k}{\pm d_i^{k)} - \rho_k} \right]_+, \quad k = 1, \ldots, M-1. \quad (3.5)$$

Given (3.3) and (3.4), it can be therefore seen that the problem turns out how to efficiently compute the values related to $d_i^{k)}$ $(i = k+1, \ldots, M)$ at the kth step of the piecewise model without solving the inverse operations explicitly.

Now, by expressing the selected pool at the kth step as $\boldsymbol{\Phi}_k = [\boldsymbol{\Phi}_{k-1},\ \mathbf{p}_k]$ and expanding the involved inverse operation, it can be easily obtained that

$$\begin{aligned}\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}} &= \boldsymbol{\Phi}_{k-1}(\boldsymbol{\Phi}_{k-1}^{\mathrm{T}}\boldsymbol{\Phi}_{k-1})^{-1}\boldsymbol{\Phi}_{k-1}^{\mathrm{T}} \\ &+\frac{\mathbf{R}_{k-1}\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k},\quad k=1,\ldots,M,\end{aligned} \tag{3.6}$$

where $\mathbf{R}_{k-1}$ is a residue matrix as also defined in [3], given by

$$\mathbf{R}_{k-1} = \mathbf{I} - \boldsymbol{\Phi}_{k-1}(\boldsymbol{\Phi}_{k-1}^{\mathrm{T}}\boldsymbol{\Phi}_{k-1})^{-1}\boldsymbol{\Phi}_{k-1}^{\mathrm{T}}. \tag{3.7}$$

By using an identity matrix $\mathbf{I}$ of size N to minus both sides of (3.6), the following also holds [3]:

$$\mathbf{R}_k = \mathbf{R}_{k-1} - \frac{\mathbf{R}_{k-1}\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k},\quad k=1,\ldots,M, \tag{3.8}$$

where $\mathbf{R}_0 = \mathbf{I} \in \Re^{N\times N}$. Assuming that there are a total of k model terms that have been included in the selected pool and $\boldsymbol{\Phi}_k = [\mathbf{p}_1,\ldots,\mathbf{p}_k]$ is of full column rank, then, we have the following facts related to $\mathbf{R}_i$, $(i=1,\ldots,k)$. First of all, geometrically, the residue matrix $\mathbf{R}_k$ is also regarded as the orthogonal projection/transformation matrix [48] for the orthogonal projection of any arbitrary vector of $\Re^N$ onto the orthogonal complement of the column space of $\boldsymbol{\Phi}_k$ (i.e., a subspace of $\Re^N$ defined by the column basis vectors in $\boldsymbol{\Phi}_k$). Correspondingly, $\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}} = \mathbf{I} - \mathbf{R}_k$ is considered as the orthogonal projection matrix for projection onto the column space of $\boldsymbol{\Phi}_k$. Hence, it is therein known that the projection matrix is idempotent ($\mathbf{R}_k^2 = \mathbf{R}_k$) and by definition of orthogonal complement $\mathbf{R}_k\boldsymbol{\Phi}_k = \mathbf{0}$ also holds (i.e., $\mathbf{R}_k\mathbf{p}_i = \mathbf{0}$ or $\mathbf{R}_k\boldsymbol{\Phi}_i = \mathbf{0}$, $i=1,\ldots,k$). Such properties of $\mathbf{R}_k$ were also deduced in [3] but in an analytical way. Moreover, the following property can then be easily obtained using (2.6) successively:

$$\begin{aligned}\mathbf{R}_k\mathbf{e}_i &= \mathbf{R}_k\mathbf{e}_{i-1} - \gamma_i\mathbf{R}_k\boldsymbol{\Phi}_i(\boldsymbol{\Phi}_i^{\mathrm{T}}\boldsymbol{\Phi}_i)^{-1}\boldsymbol{\Phi}_i^{\mathrm{T}}\mathbf{e}_{i-1} \\ &= \mathbf{R}_k\mathbf{y},\quad i=1,\ldots,k.\end{aligned} \tag{3.9}$$

where $\mathbf{e}_i$ is the model residual vector generated after introducing the ith submodel into the overall model and $\mathbf{e}_0 = \mathbf{y}$. Furthermore, the following proposition can be derived during the model selection process.

Proposition 3.1. *Given a selected model term $\mathbf{p}_i$ $(1 \le i \le k-1)$ and a model residual vector $\mathbf{e}_j$ that is generated after introducing the jth submodel $(i \le j \le k-1)$, it holds that*

$$\mathbf{p}_i^T\mathbf{R}_{i-1}\mathbf{e}_j = \mathbf{p}_i^T\mathbf{R}_{i-1}\mathbf{y}\prod_{l=i}^{j}(1-\gamma_l),\quad i \le j \le k-1,\ 1 \le i \le k-1. \tag{3.10}$$

Proof. In order to derive this proposition, first of all, using (2.6) and (3.6), the residual vector $\mathbf{e}_i$ $(i=2,\ldots,k)$ can be updated as

$$\begin{aligned}\mathbf{e}_i &= \left[\mathbf{I} - \gamma_i\boldsymbol{\Phi}_i(\boldsymbol{\Phi}_i^{\mathrm{T}}\boldsymbol{\Phi}_i)^{-1}\boldsymbol{\Phi}_i^{\mathrm{T}}\right]\mathbf{e}_{i-1} \\ &= \left[\mathbf{I} - \gamma_i\boldsymbol{\Phi}_i(\boldsymbol{\Phi}_i^{\mathrm{T}}\boldsymbol{\Phi}_i)^{-1}\boldsymbol{\Phi}_i^{\mathrm{T}}\right]\left[\mathbf{I} - \gamma_{i-1}\boldsymbol{\Phi}_{i-1}(\boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\boldsymbol{\Phi}_{i-1})^{-1}\boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\right]\mathbf{e}_{i-2} \\ &= \left[\mathbf{I} - \gamma_i\boldsymbol{\Phi}_i(\boldsymbol{\Phi}_i^{\mathrm{T}}\boldsymbol{\Phi}_i)^{-1}\boldsymbol{\Phi}_i^{\mathrm{T}} - \gamma_{i-1}(1-\gamma_i)\boldsymbol{\Phi}_{i-1}(\boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\boldsymbol{\Phi}_{i-1})^{-1}\boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\right]\mathbf{e}_{i-2} \\ &= \left\{\mathbf{I} - \left[\gamma_i + \gamma_{i-1}(1-\gamma_i)\right]\boldsymbol{\Phi}_{i-1}(\boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\boldsymbol{\Phi}_{i-1})^{-1}\boldsymbol{\Phi}_{i-1}^{\mathrm{T}} - \frac{\gamma_i\mathbf{R}_{i-1}\mathbf{p}_i\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}}{\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_i}\right\}\mathbf{e}_{i-2} \\ &= \left[1 - (\gamma_i - \frac{\gamma_i}{\gamma_{i-1}})\right]\mathbf{e}_{i-1} + (\gamma_i - \frac{\gamma_i}{\gamma_{i-1}})\mathbf{e}_{i-2} - \frac{\gamma_i\mathbf{R}_{i-1}\mathbf{p}_i\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{y}}{\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_i}.\end{aligned} \tag{3.11}$$

Now, regarding Proposition 3.1, first of all, for $i=j=1$, it can be easily found that

$$\begin{aligned}\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_j &= \mathbf{p}_1^{\mathrm{T}}\mathbf{R}_0\mathbf{e}_1\\ &= \mathbf{p}_1^{\mathrm{T}}(\mathbf{y}-\gamma_1\boldsymbol{\Phi}_1(\boldsymbol{\Phi}_1^{\mathrm{T}}\boldsymbol{\Phi}_1)^{-1}\boldsymbol{\Phi}_1^{\mathrm{T}}\mathbf{y})\\ &= (1-\gamma_1)\mathbf{p}_1^{\mathrm{T}}\mathbf{y}.\end{aligned} \tag{3.12}$$

Then, for the remaining cases, i.e., $1\le i\le k-1$, $i\le j\le k-1$, and $j\neq 1$, by using (3.11) and the geometrical properties of the residue matrix, it can be derived that

$$\begin{aligned}\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_j = [1-(\gamma_j-\frac{\gamma_j}{\gamma_{j-1}})]\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-1}+(\gamma_j-\frac{\gamma_j}{\gamma_{j-1}})\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-2}\\ -\frac{\gamma_j\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{j-1}\mathbf{p}_j\mathbf{p}_j^{\mathrm{T}}\mathbf{R}_{j-1}\mathbf{y}}{\mathbf{p}_j^{\mathrm{T}}\mathbf{R}_{j-1}\mathbf{p}_j}.\end{aligned} \tag{3.13}$$

Of these, in the case of $j=i$, by using (3.9), it is obvious that

$$\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_j = (1-\gamma_i)\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{y},\quad j=i. \tag{3.14}$$

While, in the case of $i<j\le k-1$, (3.13) becomes

$$\begin{aligned}\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_j &= [1-(\gamma_j-\frac{\gamma_j}{\gamma_{j-1}})]\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-1}+(\gamma_j-\frac{\gamma_j}{\gamma_{j-1}})\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-2}\\ &= \mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{y}\prod_{l=i}^{j}(1-\gamma_l),\quad i<j\le k-1.\end{aligned} \tag{3.15}$$

Combining all these cases, Proposition 3.1 has thus been proofed. □

As in [3], two terms ($a_{k,i}$ and b_k) are simply defined and updated as follows.

$$\begin{cases} a_{k,i} = \mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\boldsymbol{\varphi}_i = \mathbf{p}_k^{\mathrm{T}}\boldsymbol{\varphi}_i - \sum_{j=1}^{k-1} a_{j,k}a_{j,i}/a_{j,j},\quad k=1,\ldots,M,\ i=k,\ldots,M; & (3.16)\\ b_k = \mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{y} = \mathbf{p}_k^{\mathrm{T}}\mathbf{y} - \sum_{j=1}^{k-1} b_j a_{j,k}/a_{j,j},\qquad k=1,\ldots,M. & (3.17)\end{cases}$$

As a result, the term $a_{k,i}$ constitutes the kth row of the corresponding matrix $\mathbf{A}^{k)}$ with its previous $k-1$ rows being defined in the same way. Moreover, the term b_k constitutes the kth entry of vector $\mathbf{b}^{k)}$. Now, using (2.9), (3.6) and (3.9), it follows that

$$\begin{aligned} d_i^{k)} &= \boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)^{-1}\boldsymbol{\Phi}_k^{\mathrm{T}}\mathbf{e}_{k-1}\\ &= \boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_{k-1}(\boldsymbol{\Phi}_{k-1}^{\mathrm{T}}\boldsymbol{\Phi}_{k-1})^{-1}\boldsymbol{\Phi}_{k-1}^{\mathrm{T}}\mathbf{e}_{k-1} + \frac{\boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{e}_{k-1}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k}\\ &= (1-\gamma_{k-1})\boldsymbol{\varphi}_i^{\mathrm{T}}\boldsymbol{\Phi}_{k-1}(\boldsymbol{\Phi}_{k-1}^{\mathrm{T}}\boldsymbol{\Phi}_{k-1})^{-1}\boldsymbol{\Phi}_{k-1}^{\mathrm{T}}\mathbf{e}_{k-2} + \frac{\boldsymbol{\varphi}_i^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{y}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k}\\ &= (1-\gamma_{k-1})d_i^{k-1)} + \frac{a_{k,i}b_k}{a_{k,k}},\quad i=k+1,\ldots,M.\end{aligned} \tag{3.18}$$

Up to now, the step size γ_k at the kth step can be obtained as in (3.5), based on the computation of $\rho_k\in\Re$, $\mathbf{c}^{k-1)}\in\Re^{M-k+1}$ and $\mathbf{d}^{k)}\in\Re^{M-k}$. The following steps will be performed in the same way based on the recursive computation of these variables.

After a total of k model terms that have been included in the regression matrix, the final associated coefficient vector $\hat{\boldsymbol{\Theta}}_k$ for the overall piecewise model described in (2.7) will need to be computed. To start, the following holds for that each time a new model term $\mathbf{p}_j$ ($j=1,\ldots,k$)

is included into the piecewise model:

$$\begin{aligned}\mathbf{R}_j\mathbf{e}_{j-1} &= \mathbf{e}_{j-1} - \boldsymbol{\Phi}_j(\boldsymbol{\Phi}_j^{\mathrm{T}}\boldsymbol{\Phi}_j)^{-1}\boldsymbol{\Phi}_j^{\mathrm{T}}\mathbf{e}_{j-1} \\ &= \mathbf{e}_{j-1} - (\mathbf{p}_1\hat{\theta}_{1,j} + \cdots + \mathbf{p}_j\hat{\theta}_{j,j}),\end{aligned} \tag{3.19}$$

where $(\boldsymbol{\Phi}_j^{\mathrm{T}}\boldsymbol{\Phi}_j)^{-1}\boldsymbol{\Phi}_j^{\mathrm{T}}\mathbf{e}_{j-1} = [\hat{\theta}_{1,j}, \ldots, \hat{\theta}_{j,j}]^{\mathrm{T}}$. Timing both sides of the above equation by term $\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}$ for $i = j, \ldots, 1$, gives

$$\hat{\theta}_{i,j} = \frac{\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-1} - \sum_{l=i+1}^{j}\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_l\hat{\theta}_{l,j}}{\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_i}. \tag{3.20}$$

According to (2.7) and (2.8), and using (3.20), the final associated parameter $\hat{\boldsymbol{\Theta}}_k = [\hat{\Theta}_{k,1}, \ldots, \hat{\Theta}_{k,k}]^{\mathrm{T}}$ can now be obtained, with the ith entry $\hat{\Theta}_{k,i}$ $(i = 1, \ldots, k)$ being given by

$$\begin{aligned}\hat{\Theta}_{k,i} &= \sum_{j=i}^{k}\gamma_j\hat{\theta}_{i,j} \\ &= \frac{\sum_{j=i}^{k}\gamma_j\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-1} - \sum_{j=i}^{k}\sum_{l=i+1}^{j}\gamma_j\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_l\hat{\theta}_{l,j}}{\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_i} \\ &= \frac{\sum_{j=i}^{k}\gamma_j\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-1} - \sum_{l=i+1}^{k}\sum_{j=l}^{k}\gamma_j\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_l\hat{\theta}_{l,j}}{\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_i} \\ &= \frac{\sum_{j=i}^{k}\gamma_j\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-1} - \sum_{l=i+1}^{k}\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_l\hat{\Theta}_{k,l}}{\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{p}_i}.\end{aligned} \tag{3.21}$$

Using Proposition 3.1, note that

$$\begin{aligned}\sum_{j=i}^{k}\gamma_j\mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{e}_{j-1} &= \mathbf{p}_i^{\mathrm{T}}\mathbf{R}_{i-1}\mathbf{y}\sum_{j=i}^{k}\left\{\gamma_j\prod_{l=i}^{j-1}(1-\gamma_l)\right\} \\ &= \omega_i b_i,\end{aligned} \tag{3.22}$$

where $\omega_i = \sum_{j=i}^{k}\left\{\gamma_j\prod_{l=i}^{j-1}(1-\gamma_l)\right\}$, (giving $\omega_k = \gamma_k$), and it can be easily updated as

$$\omega_i = \gamma_i + (1-\gamma_i)\omega_{i+1}, \quad i = k-1, \ldots, 1. \tag{3.23}$$

As a result, the associated coefficients (3.21) for the piecewise model can be computed as

$$\hat{\Theta}_{k,i} = (\omega_i b_i - \sum_{l=i+1}^{k} a_{i,l}\hat{\Theta}_{k,l})/a_{i,i}, \quad i = k, \ldots, 1. \tag{3.24}$$

(b) The algorithm: efficient least angle regression for the construction of sparse LIP models

The efficient least angle regression for model selection of LIP models is now presented with the pseudo code described in Algorithm 2. To start, the candidate model terms $\boldsymbol{\varphi}_i$ $(i = 1, \ldots, M)$ are first generated by taking all the training samples as the potential centres of basis functions. Then, the two vectors $\mathbf{c}^{0)}$ and $\mathbf{b}^{1)}$ are initialised as $[\boldsymbol{\varphi}_1^{\mathrm{T}}\mathbf{y}, \ldots, \boldsymbol{\varphi}_M^{\mathrm{T}}\mathbf{y}]$. The first basis function $\mathbf{p}_1$ giving the largest absolute correlation for the required model output $\mathbf{y}$ is thus found, together with the assignment of $\rho_1 = \max_{i=1}^{M}|c_i^{0)}|$, $\boldsymbol{\Phi}_1 = \mathbf{p}_1$ and $k = 1$. The following loop is to recursively find the step size γ_k and the $(k+1)$th basis function $\mathbf{p}_{k+1}$ in order to include the $(k+1)$th submodel $\boldsymbol{\Phi}_{k+1}\hat{\boldsymbol{\theta}}_{k+1}$ into the whole piecewise model. To do this, the kth (row of) entries of $\mathbf{A}^{k)}$

Algorithm 2 Pseudo code for sparse LIP models construction based on the proposed efficient least angle regression

Generate candidate basis functions $\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_M$.
Initialise $\mathbf{c}^{0)} \leftarrow [\boldsymbol{\varphi}_1^{\mathrm{T}}\mathbf{y}, \ldots, \boldsymbol{\varphi}_M^{\mathrm{T}}\mathbf{y}]$ and $\mathbf{b}^{1)} \leftarrow \mathbf{c}^{0)}$.
Find the first basis function, i.e., $\mathbf{p}_1 \leftarrow \arg\max_{i=1}^{M} |c_i^{0)}|$.
Assign $\rho_1 \leftarrow \max_{i=1}^{M} |c_i^{0)}|$, $\boldsymbol{\Phi}_1 \leftarrow \mathbf{p}_1$ and $k \leftarrow 1$.
while $k \leq m$ **do**
    Update $\mathbf{A}^{k)}$, $\mathbf{b}^{k)}$ and $\mathbf{d}^{k)}$.
    Find $\mathbf{p}_{k+1} \leftarrow \arg\min_{i=k+1}^{M} [(\pm c_i^{k-1)} - \rho_k)/(\pm d_i^{k)} - \rho_k)]_+$.
    Assign $\gamma_k \leftarrow \min_{i=k+1}^{M} [(\pm c_i^{k-1)} - \rho_k)/(\pm d_i^{k)} - \rho_k)]_+$.
    Update $\boldsymbol{\Phi}_{k+1} \leftarrow [\boldsymbol{\Phi}_k,\ \mathbf{p}_{k+1}]$, $\mathbf{c}^{k)}$ and ρ_{k+1}.
    Update $k \leftarrow k + 1$.
end while
Assign $k \leftarrow m$ and $i \leftarrow k$.
while $i \geq 1$ **do**
    Update ω_i.
    Compute $\hat{\Theta}_{k,i}$.
    Update $i \leftarrow i - 1$.
end while
Output $\boldsymbol{\Phi}_k$ and $\hat{\boldsymbol{\Theta}}_k$.

and $\mathbf{b}^{k)}$ are first computed according to (3.16) and (3.17), followed by the computation of $d_i^{k)}$ $(i = k+1, \ldots, M)$ based on (3.18). The step size and the resultant basis function are then given by $\gamma_k = \min_{i=k+1}^{M} [(\pm c_i^{k-1)} - \rho_k)/(\pm d_i^{k)} - \rho_k)]_+$ and $\mathbf{p}_{k+1} = \arg\gamma_k$, respectively. The selected pool is thus updated as $\boldsymbol{\Phi}_{k+1} = [\boldsymbol{\Phi}_k,\ \mathbf{p}_{k+1}]$. In addition, the correlation for the existing basis functions in the selected pool and the remaining basis functions in the candidate pool are respectively updated as ρ_{k+1} and $c_i^{k)}$ $(i = k+1, \ldots, M)$ according to (3.3) and (3.4). This process continues until a predesignated number of basis functions have been included in the model or some criterion (e.g., Akaike information criterion (AIC)) is met. The pseudo codes presented here are illustrated by requiring a total of m model terms. Finally, the associated coefficient vector $\hat{\boldsymbol{\Theta}}_m$ for the final model can be computed according to (3.23) and (3.24) starting from calculating $\hat{\Theta}_{m,m}$ down to $\hat{\Theta}_{m,1}$.

It is worth noting that in the case of adopting the AIC method to terminate the algorithm, the sum of squared residuals (SSR) $\mathbf{e}^{\mathrm{T}}\mathbf{e}$ is required to be computed at each selection step so as to calculate the following AIC_k value with k being the size of the corresponding piecewise model.

$$AIC_k = N \log(\mathbf{e}_k^{\mathrm{T}}\mathbf{e}_k/N) + 2k \tag{3.25}$$

This can be easily realised for the proposed algorithm. According to (2.6), the SSR at each selection step can be recursively updated as

$$\mathbf{e}_k^{\mathrm{T}}\mathbf{e}_k = (1-\gamma_k)^2\mathbf{e}_{k-1}^{\mathrm{T}}\mathbf{e}_{k-1} + \gamma_k(2-\gamma_k)\mathbf{e}_{k-1}^{\mathrm{T}}\mathbf{R}_k\mathbf{e}_{k-1}, \tag{3.26}$$

where $\mathbf{e}_{k-1}^{\mathrm{T}}\mathbf{R}_k\mathbf{e}_{k-1}$ can be further updated as

$$\begin{aligned}
\mathbf{e}_{k-1}^{\mathrm{T}}\mathbf{R}_k\mathbf{e}_{k-1} &= \mathbf{e}_{k-1}^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{e}_{k-1} - \frac{\mathbf{e}_{k-1}^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{e}_{k-1}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k} \\
&= \mathbf{e}_{k-2}^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{e}_{k-2} - \frac{\mathbf{y}^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{y}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{p}_k} \\
&= \mathbf{e}_{k-2}^{\mathrm{T}}\mathbf{R}_{k-1}\mathbf{e}_{k-2} - b_k^2/a_{k,k}.
\end{aligned} \tag{3.27}$$

where terms b_k and $a_{k,k}$ are obtained after the inclusion of basis function $\mathbf{p}_k$.

(c) Computational complexity

As presented in the previous subsections, the basic arithmetic operations involved in the proposed algorithm are additions/subtractions and multiplications/divisions. Assuming that a total of N data samples together with M candidate basis functions are provided, the total additions/subtractions are

$$\begin{aligned} C_{a/s} &= mN(2M-m+1)/2+(N-1)M \\ &\quad +mM(m+9)/2-m(m^2+3m+11)/3, \end{aligned} \tag{3.28}$$

where m denotes the number of selected basis functions or model terms. On the other hand, the total multiplications/divisions are

$$\begin{aligned} C_{m/d} &= mN(2M-m+1)/2+NM \\ &\quad +mM(m+9)/2-m(m^2+3m-1)/3. \end{aligned} \tag{3.29}$$

The total computational complexity for the proposed algorithm can thus be obtained by summing (3.28) and (3.29), giving

$$\begin{aligned} C_t &= mN(2M-m+1)+(2N-1)M \\ &\quad +mM(m+9)-m(2m^2+6m+10)/3. \end{aligned} \tag{3.30}$$

Since in practical nonlinear system identification it usually follows $m \ll M \leq N$, the algorithm's major computational complexity lies on $O(2mMN-m^2(N-M)-2m^3/3)$. If the AIC method is employed to terminate the algorithm, extra computation will arise from computing the sum of squared residuals at every step, resulting in a further complexity of $2N+13m+1$, slightly more than that of the basic algorithm. It is worth mentioning that if the efficient Cholesky factorisation (based on the continuous update of the so-called Cholesky factor at each selection step) is employed to derive $\hat{\theta}_{i,j}$ and other related variables as discussed in subsection 2-(b), the corresponding computational complexity would be $O(4mMN+m^2N+m^3)$, mainly including the computation of $\boldsymbol{\Phi}_m^{\mathrm{T}}\boldsymbol{\Phi}_m$ ($O(m^2N)$), the update of Cholesky factorisation and the computation of the associated parameters for each submodel ($O(m^3+m^2N)$), and the computation of correlation and evolving direction values (in determining each step size) for candidate model terms ($O(4mMN-m^2N)$). The computational efficiency of the proposed algorithm can thus be achieved (generally saving more computational time as the selected number of model terms increases), while providing parsimonious LIP models.

4. Illustrative examples

In this section, three illustrative examples are presented to demonstrate the efficiency and effectiveness of the proposed approach. The computational efficiency and numerical stability of the proposed approach are also compared with the original approach for nonlinear system identification. The first example is for the Chaotic time-series prediction [49]; the second involves forecasting of number of sunspots [50]; and finally the third is for the classification of Australian credit approval [51]. All the experiments were conducted on a Intel(R) Core(TM)2 Duo CPU (P8600 2.40GHz), running a Windows 7 operating system, with programs executed by MATLAB.

(a) Chaotic time-series prediction

The Chaotic time-series prediction is considered in this example, the data being generated from the following Mackey-Glass time-delay differential equation:

$$\frac{dy(t)}{dt}=\frac{0.2y(t-\tau)}{1+y^{10}(t-\tau)}-0.1y(t), \tag{4.1}$$

where $\tau = 17$ and $y(0) = 1.2$ were adopted. The underlying behaviour is widely regarded as non-convergent and sensitive to initial conditions [49,52]. The proposed algorithm is used to learn this behaviour by constructing LIP models, aiming to predict the value of $y(t)$ at time instant t from the past observations of $y(t-24)$, $y(t-18)$, $y(t-12)$ and $y(t-6)$. This means that $\mathbf{x}(t) = [y(t-24), y(t-18), y(t-12), y(t-6)]$ constitutes the model input vector. As usual, the fourth-order Runge-Kutta method was applied with a step size 0.1 to solve the above equation, resulting in the selection of a total of 1000 samples covering instances from $t = 124$ to $t = 1123$ to form the whole dataset. Of these, the first half is used for training while the rest for testing.

Table 1. Training times required and the corresponding results produced by the original and proposed approaches with varied small numbers of model terms (stable cases) in example 1

# Model terms	Original (ms)	Proposed (ms)	Correlation	SSR	L_1 norm	Training RMSE	Test RMSE
m=1	19.00	16.68	2.803	12.493	1.783	1.581×10^{-1}	1.574×10^{-1}
m=2	23.97	19.32	2.393	10.132	2.238	1.424×10^{-1}	1.417×10^{-1}
m=3	28.88	22.08	2.119	8.732	2.548	1.322×10^{-1}	1.314×10^{-1}
m=4	33.84	24.93	1.417	5.464	25.408	1.045×10^{-1}	1.037×10^{-1}
m=5	39.30	27.18	1.358	5.223	27.523	1.022×10^{-1}	1.013×10^{-1}
m=10	65.18	40.37	2.803×10^{-2}	4.050×10^{-1}	1.775×10^{3}	2.846×10^{-2}	2.813×10^{-2}
m=15	90.46	53.94	4.812×10^{-3}	1.713×10^{-1}	8.375×10^{3}	1.851×10^{-2}	1.831×10^{-2}
m=20	115.69	67.77	8.435×10^{-4}	1.093×10^{-1}	1.701×10^{4}	1.479×10^{-2}	1.468×10^{-2}
m=25	143.61	82.34	1.496×10^{-4}	4.104×10^{-2}	2.645×10^{4}	9.060×10^{-3}	9.054×10^{-3}
m=30	169.59	96.44	2.826×10^{-5}	2.343×10^{-2}	4.308×10^{4}	6.845×10^{-3}	6.850×10^{-3}

The Gaussian width σ was assigned to a value of 0.7 to construct the candidate pool in which a total of 500 candidate model terms were produced at the beginning by taking all the training instances as the centres of potential basis functions. Both the original approach and our approach were used to perform the model selection, the process of selecting the first 30 model terms being shown in Table 1. Here, the original approach was realised as in [53], in which the forward and backward substitutions were adopted to successively update the Cholesky decomposition and model parameters. In Table 1, the training time, correlation, SSR, L_1 norm of model coefficients and training and test errors are listed for varied numbers of selected model terms. The model training times given in this table and the following tables in the paper were all averaged from a total of 50 runs of the respective approaches. Amongst those selected model terms, both the original and our methods were able to find the same terms to be included in the LIP model, giving the same values of correlations, SSRs, L_1 norms of coefficients and model errors. The effectiveness of our method has thus been demonstrated. In addition, the computational efficiency of the proposed approach in comparison to the original approach can be demonstrated by comparing between the second and third columns of Table 1. This superiority became more significant when more model terms were included in the constructed model.

The selected model terms from the original approach and our approach started to differ from each other at step 45, with the corresponding training times, conditions of constructed regression matrices, SSRs, L_1 norms of coefficients and model errors, respectively, shown in Table 2. Due to standardisation, a number of model terms up to $N-1$ (499 in this example) were successively included into the model in order to show the complete journey of the proposed algorithm and thus its efficiency. The condition number is seen generally increased during the model selection process as relatively more correlated terms are successively included into the model. Again, the computational efficiency of the proposed algorithm was always evident as in the third column of the table. The reason behind the differently selected model terms is mainly attributed to algorithm's stability issues, owing to the gradually ill-conditioned regression matrix being constructed. In detail, as for the original algorithm, it is recognised that at every step the Cholesky decomposition performed on the explicit formulation of $\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k$ squares the system

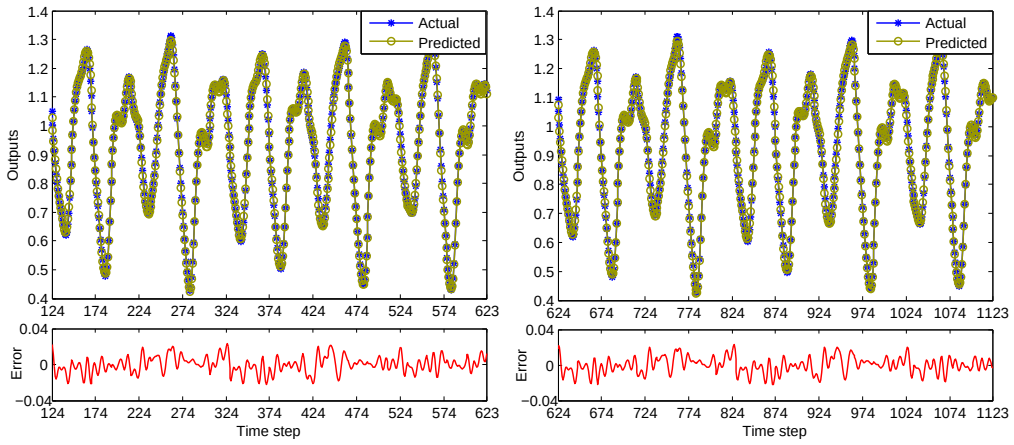
Table 2. Training times required and the corresponding results produced by the original and proposed approaches with varied large numbers of model terms (unstable cases) in example 1

# Model terms	Method	Time (ms)	Condition	SSR	L_1 norm	Training RMSE	Test RMSE
m=10	Original	65.18	8.592×10^{3}	4.050×10^{-1}	1.775×10^{3}	2.846×10^{-2}	2.813×10^{-2}
	Proposed	40.37	8.592×10^{3}	4.050×10^{-1}	1.775×10^{3}	2.846×10^{-2}	2.813×10^{-2}
m=45	Original	251.69	1.781×10^{8}	1.372×10^{-2}	1.091×10^{6}	5.238×10^{-3}	5.220×10^{-3}
	Proposed	143.30	1.315×10^{8}	1.276×10^{-2}	1.018×10^{6}	5.051×10^{-3}	5.025×10^{-3}
m=50	Original	279.57	2.504×10^{8}	9.698×10^{-3}	2.358×10^{6}	4.404×10^{-3}	4.348×10^{-3}
	Proposed	159.00	2.926×10^{8}	1.166×10^{-2}	9.156×10^{5}	4.828×10^{-3}	4.786×10^{-3}
m=100	Original	612.80	8.325×10^{11}	1.432×10^{-2}	2.473×10^{7}	5.352×10^{-3}	5.351×10^{-3}
	Proposed	338.50	2.261×10^{12}	3.490×10^{-3}	8.950×10^{6}	2.642×10^{-3}	2.617×10^{-3}
m=150	Original	1048.07	2.577×10^{14}	2.535×10^{-2}	3.798×10^{7}	7.121×10^{-3}	7.006×10^{-3}
	Proposed	550.04	5.742×10^{13}	3.230×10^{-3}	1.279×10^{7}	2.542×10^{-3}	2.507×10^{-3}
m=200	Original	1627.27	3.845×10^{15}	5.322×10^{-2}	5.792×10^{7}	1.032×10^{-2}	9.981×10^{-3}
	Proposed	756.15	1.208×10^{15}	2.926×10^{-3}	1.568×10^{7}	2.419×10^{-3}	2.392×10^{-3}
m=250	Original	2341.37	2.096×10^{16}	1.284×10^{-1}	9.070×10^{7}	1.602×10^{-2}	1.530×10^{-2}
	Proposed	1024.23	6.212×10^{15}	2.493×10^{-3}	1.477×10^{7}	2.233×10^{-3}	2.195×10^{-3}
m=300	Original	3329.18	4.970×10^{16}	2.783×10^{-1}	1.328×10^{8}	2.359×10^{-2}	2.238×10^{-2}
	Proposed	1256.22	4.443×10^{16}	2.404×10^{-3}	1.274×10^{7}	2.193×10^{-3}	2.167×10^{-3}
m=350	Original	4570.16	9.080×10^{16}	7.439×10^{-1}	2.148×10^{8}	3.857×10^{-2}	3.639×10^{-2}
	Proposed	1507.29	7.294×10^{16}	2.432×10^{-3}	1.199×10^{7}	2.205×10^{-3}	2.186×10^{-3}
m=400	Original	6056.50	1.550×10^{17}	2.067×10^{0}	3.549×10^{8}	6.430×10^{-2}	6.047×10^{-2}
	Proposed	1679.56	1.563×10^{17}	2.399×10^{-3}	1.316×10^{7}	2.190×10^{-3}	2.166×10^{-3}
m=450	Original	7862.22	3.568×10^{17}	2.388×10^{1}	1.192×10^{9}	2.185×10^{-1}	2.049×10^{-1}
	Proposed	1916.53	3.391×10^{17}	2.370×10^{-3}	1.388×10^{7}	2.177×10^{-3}	2.150×10^{-3}
m=499	Original	9902.52	1.709×10^{19}	7.235×10^{4}	6.527×10^{10}	1.203×10^{1}	1.127×10^{1}
	Proposed	2087.01	1.185×10^{19}	2.497×10^{-3}	3.118×10^{7}	2.235×10^{-3}	2.206×10^{-3}

* It is noted that the proposed approach and the original approach started to choose different model terms from model size 45 due to gradually ill-conditioned, selected regression matrices (causing potential instability issues).

condition number (i.e., $\kappa_2(\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k)=\kappa_2^2(\boldsymbol{\Phi}_k)$), where square roots are also engaged. In contrast, our method is essentially derived from the orthogonalisation of $\boldsymbol{\Phi}_k$ (wherein this does not alter the condition number, i.e., $\kappa_2(\boldsymbol{\Phi}_k)$). The corresponding orthogonal matrix is implicitly given by $\mathbf{Q}_k=[\mathbf{q}_1,\ldots,\mathbf{q}_k]$, where $\mathbf{q}_i=\mathbf{R}_{i-1}\mathbf{p}_i/||\mathbf{R}_{i-1}\mathbf{p}_i||_2$ $(i=1,\ldots,k)$. An upper triangular matrix $\mathbf{A}^{k)}$ is thereby resulted for solving the system only in the end of the model selection, plus avoiding square root operations as seen from updating all the scalars, matrices and vectors based on (3.8), i.e., ρ_{k+1}, $\mathbf{c}^{k)}$, γ_k, $\mathbf{A}^{k)}$, $\mathbf{b}^{k)}$, $\mathbf{d}^{k)}$ and $\hat{\boldsymbol{\Theta}}_m$, according to (3.3)-(3.5), (3.16)-(3.18) and (3.23)-(3.24). Consequently, the amplified condition number in the original algorithm means that the system being solved is more ill-conditioned (i.e., more sensitive to perturbations). In addition, in the original algorithm, when the selected model terms are not much correlated (i.e., matrix $\boldsymbol{\Phi}_k^{\mathrm{T}}\boldsymbol{\Phi}_k$ is positive-definite) as usually exhibited in the early stages of the model selection, the numbers under the square roots are always positive. However, with the model size increases model terms with significant correlation are then inevitably selected, resulting in increasingly ill-conditioned matrix (aka near-singular matrix). In this case, the numbers under the square roots can be negative due to roundoff and accumulated errors, causing further instability issues for the original algorithm.

From the obtained SSR and L_1 norm of coefficients for those constructed models with different sizes, it can also be seen that our method was more stable when encountered with ill-conditioned scenarios. In general, especially in the early stages of model selection, the SSR generated decreases as the model size increases. However, due to the increasing ill-conditioning of the constructed regression matrix thus causing instability issues, the SSR can then become larger when a large number of terms are included into the model. This can be seen from the results obtained by the original algorithm, confronting the large SSR increment since from a model size of 50 terms to 100 terms and thereafter. With the use of our algorithm, this instability issue caused by ill-conditioning is dramatically mitigated, where only slight increase in SSR can be seen when a large number of model terms are selected. For example, by comparing the SSR and L_1 norm produced by the original algorithm at steps 150 and 200, the instability issues apparently appeared where both SSR and L_1 norm increased dramatically (from 2.535×10^{-2} to 5.322×10^{-2}, and from 3.798×10^{7} to 5.792×10^{7}, respectively). In contrast, our method produced much better results for these two values (decreased from 3.230×10^{-3} to 2.926×10^{-3} for SSR, and increased from 1.279×10^{7} to 1.568×10^{7} for L_1 norm). The situation became extremely worse for the original approach when more model terms were to be included in the final model as shown in the bottom part of Table 2, whereas this was considerably well for our approach. The numerical stability of our algorithm in comparison with the original approach was also reflected in the resultant training and test RMSEs (root mean squared errors) as shown in the last two columns of Table 2. To retrieve a moderate size model and also prevent it from being seriously over-fitted and ill-conditioned, by adopting the AIC method as derived in (3.25)-(3.27), a total of 25 model terms can be selected to construct the LIP model for this example. Fig. 1 depicts the actual and predicted outputs of the system over the training and test datasets, showing a good representation of the chaotic behaviours for the model developed. The resultant training and test RMSEs are thereby 9.060×10^{-3} and 9.054×10^{-3}, respectively.



(a) Training output (b) Test output

Figure 1. The training and test outputs in chaotic time-series prediction (In each figure, the upper subfigure depicts the actual and predicted outputs, while the bottom subfigure shows the error between them).

(b) Number of sunspots forecasting

The number of sunspots being used to indicate the abundance of sunspots on the Sun, is recognised exhibiting nonlinear, non-stationary and non-Gaussian behaviours [54]. In this example, the yearly mean total number of sunspots from year 1700 to 2014 was adopted from WDC-SILSO (World Data Center - Sunspot Index and Long-term Solar Observations) [50]. The proposed method was thus applied to develop a LIP model to forecast the number of sunspots.

As in previous studies [54], the input vector for the model were defined as $\mathbf{x}(t) = [y(t-3), y(t-2), y(t-1)]$, where t denotes the year index, and the output was to forecast $y(t)$ of the sunspot number at year t. The sunspot numbers for the first 156 years (i.e., $1703 \leq t \leq 1858$) were used for training, with the trained model then applied to forecast the number of sunspots for the remaining 156 years (i.e., $1859 \leq t \leq 2014$).

Table 3. Training times required and the corresponding results produced by the original and proposed approaches with varied numbers of model terms in example 2

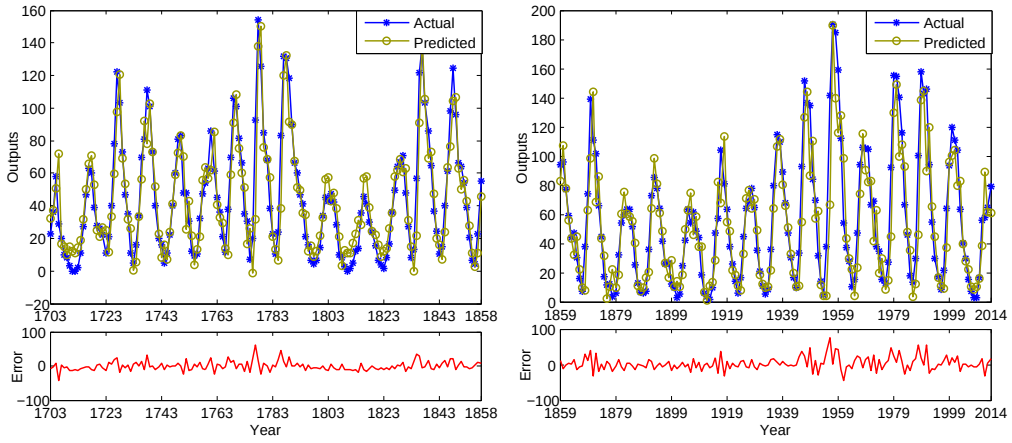
# Model terms	Method	Time (ms)	Condition	SSR	L_1 norm	Training RMSE	Test RMSE
m=1	Original	0.71	1.000×10^{0}	1.297×10^{5}	1.097×10^{2}	28.8346	40.0292
	Proposed	0.70	1.000×10^{0}	1.297×10^{5}	1.097×10^{2}	28.8346	40.0292
m=5	Original	1.98	8.344×10^{2}	2.863×10^{4}	5.990×10^{4}	13.5468	19.0125
	Proposed	1.54	8.344×10^{2}	2.863×10^{4}	5.990×10^{4}	13.5468	19.0125
m=10	Original	3.89	5.488×10^{5}	2.404×10^{4}	6.086×10^{6}	12.4148	18.0126
	Proposed	2.66	5.488×10^{5}	2.404×10^{4}	6.086×10^{6}	12.4148	18.0126
m=20	Original	9.44	3.351×10^{9}	2.057×10^{4}	6.091×10^{9}	11.4840	27.9171
	Proposed	5.89	4.933×10^{9}	2.076×10^{4}	2.256×10^{9}	11.5348	26.1846
m=50	Original	42.12	1.851×10^{15}	2.115×10^{4}	1.373×10^{10}	11.6434	35.1763
	Proposed	20.02	3.776×10^{15}	2.025×10^{4}	6.631×10^{9}	11.3936	23.4977
m=100	Original	151.78	8.512×10^{15}	6.380×10^{4}	6.555×10^{10}	20.2234	100.9686
	Proposed	57.43	9.669×10^{15}	1.967×10^{4}	8.602×10^{9}	11.2280	19.7732
m=150	Original	347.67	1.175×10^{17}	2.009×10^{6}	4.046×10^{11}	113.4860	566.4108
	Proposed	104.55	1.012×10^{17}	1.967×10^{4}	1.429×10^{10}	11.2292	20.0788
m=155	Original	372.89	2.504×10^{17}	1.282×10^{7}	1.016×10^{12}	286.6082	1.409×10^{3}
	Proposed	112.35	1.861×10^{17}	1.966×10^{4}	1.895×10^{10}	11.2265	19.9704

* It is noted that the proposed approach and the original approach started to choose different model terms from model size 20 due to gradually ill-conditioned, selected regression matrices (causing potential instability issues).

As in example 1, the Gaussian width σ was assigned to 600 in this example and a total of 156 candidate model terms were generated. The training times, conditions of selected regression matrices, SSRs, L_1 norms and model errors under a varied number of selected model terms are listed in Table 3 for both the original and proposed approaches. It can be found that the training times consumed by our method were less than those consumed by the original approach, and this superiority was seen more significant as the number of included model terms increased. Meanwhile, our algorithm was able to perform same results as in the original approach when a number of up to 20 model terms were selected.

Due to ill-conditioning issues, it was found that, in comparison with the original approach, the proposed algorithm started to choose differing model terms since model size of 20. As the number of model terms as well as the condition of selected regression matrix increased, the proposed approach showed dramatic numerical stability than the original approach. When a total of 155 model terms were selected, the SSRs resulting from the proposed and original approaches were 1.966×10^{4} and 1.282×10^{7}, respectively, while the corresponding L_1 norms were 1.895×10^{10} and 1.016×10^{12}, respectively. Correspondingly, the training and test RMSEs from our algorithm were also found to be much better than from the original approach. For the original approach, due to instability issues, the model errors continued to increase after a certain number of model terms have been included in the model. The overfitting issues, however, can somewhat be observed from this example, although it is not obvious for our method. A subset of five model terms can be

adopted by using the AIC method, the results being depicted in Fig. 2. The corresponding RMSEs on the training and test dataset were 13.5468 and 19.0125, respectively.



(a) Training output (b) Test output

Figure 2. The training and test outputs in forecasting the number of sunspots (In each figure, the upper subfigure depicts the actual and predicted outputs, while the bottom subfigure shows the error between them).

(c) Australian credit approval

In this example, the Statlog dataset consisting of the information of Australian credit card applications obtained from the UCI machine learning repository [51] is considered. For privacy protection, the names and values of attributes for the dataset were initially altered to be physically meaningless, where a total of 14 input attributes are included. The output attribute is used to indicate either the approval or the refusal of a credit card application. There are 690 instances in the dataset, from which 37 instances involve missing values (replaced with artificially computed values). The whole dataset was then randomly and equally partitioned into a training and a test dataset for this study.

The Gaussian width σ was chosen as 5.0×10^7 and a total of 345 candidate model terms were obtained. As in examples 1 and 2, the training times, conditions of selected regression matrices, SSRs, L_1 norms and model accuracies for various subsets of selected model terms for both the original and proposed approaches are given in Table 4. The superiority of our approach in terms of consuming less training time and providing more stable results was once again confirmed. In this example, differing model terms started to be chosen since step 11 from the two approaches, resulting in a remarkable difference between them when a total of 344 model terms were included into the model (SSR: 2.125×10^2 and 1.139×10^6; L_1 norm: 2.435×10^9 and 1.916×10^{11}). After a few model terms that have been selected, the test accuracy has stabilised around 80% for our approach while this is not the case for the original approach. A total of six model terms can be chosen to construct the final classifier for this example by using the AIC method, giving the training and test accuracies of 77.97% and 80.58%, respectively.

5. Conclusion

An efficient least angle regression algorithm was proposed in this paper for the construction of a class of LIP (linear-in-the-parameters) models. The algorithm proceeds in a stepwise manner, each step adding a new model term into the LIP model, with the effect of minimising model residuals and L_1 norm of coefficients. It is a less greedy model selection method and proceeds

Table 4. Training times required and the corresponding results produced by the original and proposed approaches with varied numbers of model terms in Example 3

# Model terms	Method	Time (ms)	Condition	SSR	L_1 norm	Training acc.	Test acc.
m=1	Original	7.35	1.000×10^{0}	3.274×10^{2}	1.873×10^{0}	58.55%	60.29%
	Proposed	6.63	1.000×10^{0}	3.274×10^{2}	1.873×10^{0}	58.55%	60.29%
m=5	Original	16.20	2.151×10^{6}	2.554×10^{2}	6.157×10^{6}	73.04%	73.04%
	Proposed	10.88	2.151×10^{6}	2.554×10^{2}	6.156×10^{6}	73.04%	73.04%
m=11	Original	30.17	6.922×10^{8}	2.219×10^{2}	1.077×10^{8}	79.13%	80.00%
	Proposed	17.63	7.207×10^{8}	2.211×10^{2}	1.011×10^{8}	79.42%	79.42%
m=50	Original	141.31	1.245×10^{10}	2.152×10^{2}	4.199×10^{8}	79.42%	79.71%
	Proposed	73.40	1.228×10^{10}	2.280×10^{2}	2.907×10^{8}	78.55%	78.55%
m=100	Original	347.38	2.368×10^{10}	2.145×10^{2}	6.785×10^{8}	80.29%	81.16%
	Proposed	166.87	2.316×10^{10}	2.207×10^{2}	4.995×10^{8}	78.84%	79.71%
m=150	Original	640.86	3.983×10^{10}	2.175×10^{2}	9.202×10^{8}	78.84%	78.55%
	Proposed	282.42	3.879×10^{10}	2.187×10^{2}	6.077×10^{8}	79.13%	80.29%
m=200	Original	1004.45	6.536×10^{10}	2.292×10^{2}	1.287×10^{9}	76.23%	77.10%
	Proposed	396.01	6.190×10^{10}	2.185×10^{2}	7.637×10^{8}	79.42%	80.58%
m=250	Original	1586.01	1.278×10^{11}	2.817×10^{2}	2.065×10^{9}	71.88%	71.59%
	Proposed	521.28	1.139×10^{11}	2.194×10^{2}	1.151×10^{9}	78.84%	80.58%
m=300	Original	2429.72	2.623×10^{11}	5.736×10^{2}	3.988×10^{9}	70.14%	73.04%
	Proposed	643.63	2.574×10^{11}	2.109×10^{2}	1.722×10^{9}	79.71%	81.45%
m=344	Original	3395.23	1.057×10^{13}	1.139×10^{6}	1.916×10^{11}	62.90%	64.35%
	Proposed	744.85	1.132×10^{13}	2.125×10^{2}	2.435×10^{9}	78.84%	79.42%

* It is noted that the proposed approach and the original approach started to choose different model terms from model size 11 due to gradually ill-conditioned, selected regression matrices (causing potential instability issues).

equiangularly between all the selected model terms by means of their resultant correlations with model errors at every subset selection step. Unlike the original approach where the well-known Cholesky decomposition was employed, an efficient recursive algorithm was proposed to solve the least angle regression without the need of matrix inversions, decompositions and transformations. The correlations between model terms and residuals, the evolving directions, together with other pertinent variables, were explicitly formulated and are recursively updated in the proposed algorithm. The model coefficients are only computed when the algorithm finishes. The computational complexity of the proposed approach was also well analysed and confirmed to be more efficient than the original approach. Finally, three artificial and real-world illustrative examples were used to demonstrate the effectiveness and efficiency of the proposed algorithm, where its numerical stability was also found to be another strength. As mentioned previously, given the connection between LAR and Lasso, future research includes deriving the efficient algorithm for retrieving Lasso solutions for performing tasks such as model or variable selection. On the other hand, the integration of hyperparameter learning for enhanced model performance is another direction of research.

Ethics. This research does not contain human or animal subject.

Data Accessibility. All the experimental data are accessible from the references cited in the correponding examples in the paper.

Authors' Contributions. W.Z. developed the algorithm, conducted the simulation and drafted the manuscript. T.H.B. and Y.R. coordinated the research, helped to interpret the results and revised the manuscript. All authors gave final approval for publication.

Competing Interests. We have no competing interests.

Funding. The work was supported by the EU Seventh Framework Programme (FP7) under Grant 619795.

Acknowledgements. The authors acknowledge the editors and reviewers for their constructive comments and suggestions for improving the quality of this paper.

References

1. Hong X, Harris CJ, Chen S, Sharkey PM. 2003 Robust nonlinear model identification methods using forward regression. *IEEE Trans. Syst., Man, Cybern. A, Syst. Hum.* **33**, 514–523. (doi:10.1109/TSMCA.2003.809217)
2. Billings SA, Wei HL. 2007 Sparse model identification using a forward orthogonal regression algorithm aided by mutual information. *IEEE Trans. Neural Netw.* **18**, 306–310. (doi:10.1109/TNN.2006.886356)
3. Li K, Peng JX, Irwin GW. 2005 A fast nonlinear model identification method. *IEEE Trans. Autom. Control* **50**, 1211–1216. (doi:10.1109/TAC.2005.852557)
4. Han HG, Qiao JF. 2012 Adaptive computation algorithm for RBF neural network. *IEEE Trans. Neural Netw. Learn. Syst.* **23**, 342–347. (doi:10.1109/TNNLS.2011.2178559)
5. Wang L, Truong DN. 2013 Stability enhancement of a power system with a PMSG-based and a DFIG-based offshore wind farm using a SVC with an adaptive-network-based fuzzy inference system. *IEEE Trans. Ind. Electron.* **60**, 2799–2807. (doi:10.1109/TIE.2012.2218557)
6. Wu X, Różycki P, Wilamowski BM. 2015 A hybrid constructive algorithm for single-layer feedforward networks learning. *IEEE Trans. Neural Netw. Learn. Syst.* **26**, 1659–1668. (doi:10.1109/TNNLS.2014.2350957)
7. Billings SA, Chen S, Korenberg MJ. 1989 Identification of MIMO non-linear systems using a forward-regression orthogonal estimator. *Int. J. Control* **49**, 2157–2189. (doi:10.1080/00207178908961377)
8. Aguirre LA. 1997 On the structure of nonlinear polynomial models: higher order correlation functions, spectra, and term clusters. *IEEE Trans. Circuits Syst. I, Fundamental Theory Appl.* **44**, 450–453. (doi:10.1109/81.572342)
9. Pukish MS, Różycki P, Wilamowski BM. 2015 Polynet: A polynomial-based learning machine for universal approximation. *IEEE Trans. Ind. Inf.* **11**, 708–716. (doi:10.1109/TII.2015.2426012)
10. Er MJ, Wu S, Lu J, Toh HL. 2002 Face recognition with radial basis function (RBF) neural networks. *IEEE Trans. Neural Netw.* **13**, 697–710. (doi:10.1109/TNN.2002.1000134)
11. Chen H, Gong Y, Hong X. 2013 Online modeling with tunable RBF network. *IEEE Trans. Cybern.* **43**, 935–947. (doi:10.1109/TSMCB.2012.2218804)
12. Yu H, Reiner PD, Xie T, Bartczak T, Wilamowski BM. 2014 An incremental design of radial basis function networks. *IEEE Trans. Neural Netw. Learn. Syst.* **25**, 1793–1803. (doi:10.1109/TNNLS.2013.2295813)
13. Chiang JH, Hao PY. 2004 Support vector learning mechanism for fuzzy rule-based modeling: A new approach. *IEEE Trans. Fuzzy Syst.* **12**, 1–12. (doi:10.1109/TFUZZ.2003.817839)
14. Juang CF, Hung CW, Hsu CH. 2014 Rule-based cooperative continuous ant colony optimization to improve the accuracy of fuzzy system design. *IEEE Trans. Fuzzy Syst.* **22**, 723–735. (doi:10.1109/TFUZZ.2013.2272480)
15. Zhao W, Zhang J, Li K. 2015 An efficient LS-SVM-based method for fuzzy system construction. *IEEE Trans. Fuzzy Syst.* **23**, 627–643. (doi:10.1109/TFUZZ.2014.2321594)
16. Zhang Y, Liu Y. 2009 Data imputation using least squares support vector machines in urban arterial streets. *IEEE Signal Process. Lett.* **16**, 414–417. (doi:10.1109/LSP.2009.2016451)
17. Yu L, Chen H, Wang S, Lai KK. 2009 Evolving least squares support vector machines for stock market trend mining. *IEEE Trans. Evol. Comput.* **13**, 87–102. (doi:10.1109/TEVC.2008.928176)
18. Zhang J, Li K, Irwin GW, Zhao W. 2012 A regression approach to LS-SVM and sparse realization based on fast subset selection. in *Proc. 10th Word Congr. Intell. Control Autom.* Beijing 612–617 (doi:10.1109/WCICA.2012.6357952)

19. Chen S, Billings SA, Grant PM. 1992 Recursive hybrid algorithm for non-linear system identification using radial basis function networks. *Int. J. Control* **55**, 1051–1070. (doi:10.1080/00207179208934272)
20. Guyon I, Elisseeff A. 2003 An introduction to variable and feature selection. *J. Mach. Learn. Res.* **3**, 1157–1182. (doi:10.1162/153244303322753616)
21. Kasabov NK, Song Q. 2002 DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE Trans. Fuzzy Syst.* **10**, 144–154. (doi:10.1109/91.995117)
22. Pham DT, Suarez-Alvarez MM, Prostov YI. 2011 Random search with k-prototypes algorithm for clustering mixed datasets. *Proc. R. Soc. A* **467**, 2387–2403. (doi:10.1098/rspa.2010.0594)
23. Yen J, Wang L. 1998 Application of statistical information criteria for optimal fuzzy model construction. *IEEE Trans. Fuzzy Syst.* **6**, 362–372. (doi:10.1109/91.705503)
24. Setnes M, Babuska R. 2001 Rule base reduction: some comments on the use of orthogonal transforms. *IEEE Trans. Syst., Man, Cybern.* **31**, 199–206. (doi:10.1109/5326.941843)
25. Hastie T, Tibshirani R, Friedman J. 2009 *The elements of statistical learning: data mining, inference, and prediction.* Springer-Verlag New York.
26. Chen S, Billings SA, Luo W. 1989 Orthogonal least squares methods and their application to non-linear system identification. *Int. J. Control* **50**, 1873–1896. (doi:10.1080/00207178908953472)
27. Wang LX, Mendel JM. 1992 Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. *IEEE Trans. Neural Netw.* **3**, 807–814. (doi:10.1109/72.159070)
28. Stark J. 1997 Adaptive model selection using orthogonal least squares methods. *Proc. R. Soc. Lond. A* **453**, 21–42. (doi:10.1098/rspa.1997.0002)
29. Hong X, Harris CJ. 2001 Variable selection algorithm for the construction of MIMO operating point dependent neurofuzzy networks. *IEEE Trans. Fuzzy Syst.* **9**, 88–101. (doi:10.1109/91.917117)
30. Chen S, Hong X, Luk BL, Harris CJ. 2009 Orthogonal-least-squares regression: a unified approach for data modelling. *Neurocomputing* **72**, 2670–2681. (doi:10.1016/j.neucom.2008.10.002)
31. Zhao W, Niu Q, Li K, Irwin GW. 2013 A hybrid learning method for constructing compact rule-based fuzzy models. *IEEE Trans. Cybern.* **43**, 1807–1821. (doi:10.1109/TSMCB.2012.2231068)
32. Efron B, Hastie T, Johnstone I, Tibshirani R. 2004 Least angle regression. *Ann. Stat.* **32**, 407–451. (doi:10.1214/009053604000000067)
33. Hastie T, Taylor J, Tibshirani R, Walther G. 2007 Forward stagewise regression and the monotone lasso. *Electron. J. Stat.* **1**, 1–29. (doi:10.1214/07-EJS004)
34. Chen S. 2006 Local regularization assisted orthogonal least squares regression. *Neurocomputing* **69**, 559–585. (doi:10.1016/j.neucom.2004.12.011)
35. Chen S, Hong X, Harris CJ. 2008 An orthogonal forward regression technique for sparse kernel density estimation. *Neurocomputing* **71**, 931–943. (doi:10.1016/j.neucom.2007.02.008)
36. Hong X, Chen S. 2015 Elastic net orthogonal forward regression. *Neurocomputing* **148**, 551–560. (doi:10.1016/j.neucom.2014.07.008)
37. Li K, Peng JX, Bai EW. 2006 A two-stage algorithm for identification of nonlinear dynamic systems. *Automatica* **42**, 1189–1197. (doi:10.1016/j.automatica.2006.03.004)
38. Xiao W, Wu Y, Zhou H. 2015 ConvexLAR: an extension of least angle regression. *J. Comput. Graph. Stat.* **24**, 603–626. (doi:10.1080/10618600.2014.962700)
39. Vidyasagar M. 2014 Machine learning methods in the computational biology of cancer. *Proc. R. Soc. A* **470** (doi:10.1098/rspa.2014.0081)
40. Augugliaro L, Mineo AM, Wit EC. 2013 Differential geometric least angle regression: a differential geometric approach to sparse generalized linear models. *J. R. Statist. Soc. B* **75**, 471–498. (doi:10.1111/rssb.12000)
41. Wu Y. 2011 An ordinary differential equation based solution path algorithm. *J. Nonparametr. Stat.* **23**, 185–199. (doi:10.1080/10485252.2010.490584)
42. Wu Y. 2012 Elastic net for Cox's proportional hazards model with a solution path algorithm. *Stat. Sin.* **22**, 271–294. (doi:10.5705/ss.2010.107)
43. Liu C, Yang SX, Deng L. 2015 A comparative study for least angle regression on NIR spectra analysis to determine internal qualities of navel oranges. *Expert Syst. Appl.* **42**, 8497–8503. (doi:10.1016/j.eswa.2015.07.005)

44. Hesterberg T, Choi NH, Meier L, Fraley C. 2008 Least angle and ℓ_1 penalized regression: A review. *Stat. Surv.* **2**, 61–93. (doi:10.1214/08-SS035)
45. Billings SA. 2013 *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains.* John Wiley & Sons.
46. Chetwynd D, Worden K, Manson G. 2006 An application of interval-valued neural networks to a regression problem. *Proc. R. Soc. A* **462**, 3097–3114. (doi:10.1098/rspa.2006.1717)
47. Lee JD, Sun DL, Sun Y, Taylor JE. 2016 Exact post-selection inference, with application to the lasso. *Ann. Stat.* **44**, 907–927. (doi:10.1214/15-AOS1371)
48. Seber GAF. 2008 *A matrix handbook for statisticians.* John Wiley & Sons.
49. Juang CF, Hsiao CM, Hsu CH. 2010 Hierarchical cluster-based multispecies particle-swarm optimization for fuzzy-system optimization. *IEEE Trans. Fuzzy Syst.* **18**, 14–26. (doi:10.1109/TFUZZ.2009.2034529)
50. Royal Observatory of Belgium, Ringlaan 3, 1180 Brussel, Belgium. 1700-2014 SILSO World Data Center — Sunspot Number and Long-term Solar Observations. URL http://sidc.oma.be/silso/datafiles accessed December 23, 2016
51. Bache K, Lichman M. 2013 UCI machine learning repository. URL http://archive.ics.uci.edu/ml
52. The MathWorks, Inc, Chaotic time-series prediction. URL http://uk.mathworks.com/help/fuzzy/examples/chaotic-time-series-prediction.html accessed December 23, 2016
53. Sjöstrand K. 2005 Matlab implementation of LASSO, LARS, the elastic net and SPCA. Informatics and Mathematical Modelling, Technical University of Denmark, DTU.
54. Ling SH, Leung FHF, Lam HK, Lee YS, Tam PKS. 2003 A novel genetic-algorithm-based neural network for short-term load forecasting. *IEEE Trans. Ind. Electron.* **50**, 793–799. (doi:10.1109/TIE.2003.814869)